# Training Noise-Robust Deep Neural Networks via Meta-Learning

Zhen Wang*[1], Guosheng Hu*[2], Qinghua Hu†[1]
[1]Tianjin Key Lab of Machine Learning, College of Intelligence and Computing, Tianjin University, Tianjin, China [2]AnyVision
wangzhen315@tju.edu.cn, huguosheng100@gmail.com, huqinghua@tju.edu.cn

## Abstract

*Label noise may significantly degrade the performance of Deep Neural Networks (DNNs). To train noise-robust DNNs, Loss correction (LC) approaches have been introduced. LC approaches assume the noisy labels are corrupted from clean (ground-truth) labels by an unknown noise transition matrix $T$. The backbone DNNs and $T$ can be trained separately, where $T$ is approximated by prior knowledge. For example, $T$ can be constructed by stacking the maximum or mean predictions of the samples from each class. In this work, we propose a new loss correction approach, named as Meta Loss Correction (MLC), to directly learn $T$ from data via the meta-learning framework. The MLC is model-agnostic and learns $T$ from data rather than heuristically approximates $T$ using prior knowledge. Extensive evaluations are conducted on computer vision (MNIST, CIFAR-10, CIFAR-100, Clothing1M) and natural language processing (Twitter) datasets. The experimental results show that MLC achieves very competitive performance against state-of-the-art approaches.*

## 1. Introduction

Deep learning has achieved great success on computer vision tasks such as object detection [9], image classification [20], segmentation [2], face recognition [13]. It is well known that the performance of DNNs highly relies on the large-scale high quality well-labeled training data. However, collecting such big clean data is expensive and time-consuming. To collect such data, people usually turn to search engine, automatic tagging software and crowdsourcing, which inevitably bring label noises (wrong or corrupted labels). The label noises can lead the DNNs to overfit to such noises [38], eventually degrading the model generalization performance.

The loss correction (LC) approaches [17, 28, 12] recently achieved great success on noise-robust deep learning. LC approaches assume the noisy labels are corrupted from clean (ground-truth) labels by an unknown noise transition matrix $T$. Thus, the LC approaches try to learn this matrix accurately. Some early works [31, 17] add a linear noise layer at the end of backbone Convolutional Neural Network (CNNs) to *implicitly* estimate matrix $T$. Unlike these implicit optimizations, the LC approaches [28, 12] *explicitly* estimate $T$. For example, a 'perfect example' assumption [28] is made to approximate 'perfect example' as the one with the maximum prediction in each class. Then $T$ is estimated by stacking the prediction of each 'perfect example'. Instead of using the maximum predictions, Gold Loss Correction (GLC) [12] uses the mean predictions of a small clean dataset to estimate $T$. Clearly, these approaches [28, 12], which use prior knowledge to estimate $T$, are heuristic and the 'perfect example' assumption cannot always hold true.

To learn $T$ directly from data rather than in a heuristic way, we introduce meta-learning. Meta-learning is a general data-driven optimization framework, and it can learn experience (meta-parameters) from data (meta-data). More general, meta-parameters can be some parameters to be optimized in deep learning. Recently, meta-learning achieved great success on many optimization tasks including: hyper-parameter optimization [18], neural architecture searching [41] and optimizer selection [26], etc. Most meta-learning approaches contain two optimization loops: an *inner* loop (Actual-Train) conducts the main optimization (e.g. the main deep network training), and an *outer* loop (Meta-Train) optimizes some aspects (meta-parameters, e.g. hyper-parameters of the main network) of *inner* loop.

Motivated by the success of meta-parameter optimization, in this work, we adapt meta-learning to optimize $T$ by viewing $T$ as the meta-parameter. Meta-learning usually uses a small 'clean' validation set to conduct *outer* loop optimization (Meta-Train) [30] . A small 'clean' validation set is also a popular setting for LC noise-robust learning approach [12]. With meta-learning and a small valida-

* Indicates equal contributions.
† Corresponding author: Qinghua Hu (huqinghua@tju.edu.cn)

tion set, we propose a new loss correction approach, Meta Loss Correction (MLC), to learn noise transition matrix $T$. Specifically, we conduct an alternating optimization to optimize $T$ and main (backbone) network weights $\theta$. First, we make one-step-forward virtual optimization of $\theta$ on the noisy training set during Virtual-Train; During Meta-Train, we then optimize $T$ (meta-parameter) guided by the loss (meta-objective) on validation set with the one-step-forward $\theta$ fixed; Finally we optimize the unrolled $\theta$ with the updated $T$ on the noisy training set in the stage of Actual-Train.

Our contributions can be summarized as follows: We propose a new loss correction approach, Meta Loss Correction (MLC), to learn noise transition matrix. MLC is model-agnostic and can adapt to different backbone networks and can easily generalize to tasks on both computer vision (CV) and natural language processing (NLP). Our MLC does not rely on the 'perfect example' assumption [28] and learns $T$ from data rather than directly uses prior knowledge (stacking the maximum predictions [28] or mean predictions [12] to construct $T$). We conduct extensive evaluations on CV datasets: MNIST [3], CIFAR-10 [19], CIFAR-100 [19], Clothing1M [35] and NLP dataset Twitter [8]. MLC achieves very competitive performance on these datasets over state-of-the-art approaches.

The paper is organized as follows. In the next section we present a brief introduction to related work. Our methodology is introduced in Section 3. The proposed algorithm is evaluated in Section 4. Section 5 draws the paper to a conclusion.

## 2. Related work

In this section, we briefly review the existing research on label noise.

**Robust loss function** is widely investigated. [27] proposes two robust loss functions to deal with aerial noise, one handles asymmetric omission noise and the other models registration errors, which can be optimized via an EM algorithm. [29] introduces 'soft' and 'hard' loss functions based on bootstrapping. [7] proves a sufficient condition under that loss to be tolerant to uniform label noise, and shows that 0-1 loss and sigmoid loss satisfy that condition. Then, [6] further proves that mean absolute error (MAE) is a noise-robust loss for deep CNN. Then, [40] proposes a generalized CrossEntropy (GCE) loss, which is a generalization of MAE and traditional CrossEntropy (CE) loss. The weights between MAE and CE can be adapted by tuning the parameters of GCE. Robust loss functions achieve some success, however, they cannot perform well on challenging noisy datasets.

**Relabelling** is the re-assignment of the labels for noisy samples. Relabeling includes two settings: (1) including a small clean dataset and (2) no such a dataset. For (1), [32] proposes a multi-task network: one cleaning model is trained on clean samples to clean (relabel) noisy data, and then one classification model is trained on a merged dataset (clean and relabeled data). [22] distills information from knowledge graph and clean labels to guide the relabelling of the noisy data. For (2), [25] introduces a self-error-correcting (SEC) strategy to relabel the noisy data based on the prediction/confidence of a CNNs. [34] also uses the predictions to relabel the samples.

**Weighting** aims to learn to assign small weights to samples with corrupted labels. [11] introduces co-teaching strategy which simultaneously trains two networks. These two networks select training samples with small loss (expected clean samples) and then communicate with each other with those selected samples for training. Self-paced learning [15, 39] is proposed to learn the weights of training samples guided by the training loss. For random classification noise, [24] designs an importance reweighting method to reweight samples by employing the in-versed noise rates. [16] employs an additional LSTM network to learn the optimal weights of training samples. [30] reweights the training samples by employing a small validation set. [33] detects noisy labels and reweights the noisy samples based on the confidence supplied by noisy label detection.

**Loss correction approach** recently achieved great success on noise-robust learning. Basically, the noise transition matrix $T$ is introduced to correct the predictions. Then the approaches of this category aim to learn the optimal $T$ which can lead to noise-robust performance. [31, 1] add an extra linear layer at the end of a backbone CNN that simulates the noise transition matrix. Instead of modifying architecture, [28, 12, 10] use prior knowledge to estimate $T$ e.g., via stacking the maximum [28] and mean predictions [12] of each class from samples. Our work belongs to this category. As aforementioned, most existing approaches estimate $T$ based on prior knowledge. In comparison, we directly optimize $T$ from data without relying on prior knowledge and assumptions.

## 3. Methodology

### 3.1. Noisy Label Problem

In many applications, the collected dataset is corrupted by label noises. Denote the noisy dataset by $D_\eta = \{(x_i, \tilde{y}_i), 1 \leq i \leq N\}$ where $\tilde{y}_i \in \{0,1\}^C$ may be noisy label in $C$ classes. Denote the $C \times C$ noise transition matrix by $T$ which specifies the probability of clean label $i$ flipping to noisy label $j$ by $T_{ij} = p(\tilde{y} = j | y = i)$. Following [12, 30], it is assumed that we have access to a small clean dataset. It is a sound assumption since it is feasible to collect such a dataset in the real world. Denote the small clean dataset (which usually works as validation set) by $D_v = \{(x_i, y_i), 1 \leq i \leq M\}$, $M \ll N$.

Let $f(x; \theta)$ denote the backbone DNN which is encoded

by $\theta$, then the CrossEntropy (CE) loss is expressed as:

$$Loss = CE(f(x;\theta),\tilde{y}) = -\frac{1}{N}\sum_{i=1}^{N}\tilde{y}_i \log(f(x_i;\theta)) \quad (1)$$

Given the noise transition matrix $T$, we modify the loss function Eq. (1) to include $T$ to achieve noise robust model training. Thus, the corrected loss function is represented as:

$$Loss_{LC} = -\frac{1}{N}\sum_{i=1}^{N}\tilde{y}_i \log(Tf(x_i;\theta)) \quad (2)$$

**Existing approaches** The effectiveness of loss correction approaches highly depends on the estimate of $T$. To estimate $T$, [28] makes the assumption that there exists 'perfect example' $x_i'$ of each class $i$ that $p(y=i|x_i')=1$. Then 'perfect example' $x_i'$ is approximated by the sample which has the maximum prediction/probability (softmax score) of class $i$. Then $T_{ij} = p(y=j|x_i')$. Unlike [28], Gold Loss Correction (GLC) [12] approximates $T$ using the mean prediction of all the samples belonging to class $i$ on a clean validation set instead of the maximum prediction.

**Motivation** Although [28, 12] achieve promising performance, the assumptions in [28] and [12] cannot always hold true. For example, we cannot guarantee the 'perfect example' of each class always exists. In addition, the estimation of $T$ is heuristic, because $T$ is constructed directly by the simple operations, i.e. maximum [28] or mean[12] of the predictions of samples. In this work, we propose a learning-based model that learns $T$ by employing a meta-learning optimization strategy, Meta Loss Correction (MLC). Our MLC does not rely on 'perfect example' assumption or approximate $T$ by the predictions of samples. Instead, MLC optimizes $T$ directly from data.

### 3.2. Optimizing Transition Matrix $T$ via Meta-learning

In this work, we conduct an alternating optimization to optimize noise transition matrix $T$ and the backbone network encoded by $\theta$ via the Meta Loss Correction (MLC) strategy. Specifically, the MLC approach contains three stages: Virtual-Train, Meta-Train and Actual-Train. Alternating optimization is performed on these three stages. During *Virtual-Train* stage, we optimize the backbone network weights $\theta^t$ to obtain $\hat{\theta}^{t+1}$ with a fixed $T^t$ (which is optimized in the previous iteration) guided by the corrected loss function on noisy training set $D_\eta$. Note that this is a 'virtual' step, meaning that the backbone network does not actually move to $\hat{\theta}^{t+1}$. The 'virtual' step makes preparations for estimating $T^{t+1}$ in the next stage. During *Meta-Train* stage, we optimize $T^{t+1}$ by keeping $\hat{\theta}^{t+1}$ fixed under CrossEntropy loss on a small clean validation set $D_v$. The motivation of *Meta-Train* is that we would like to find a

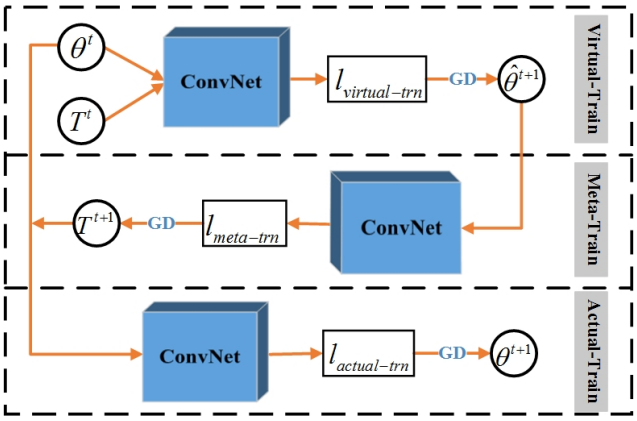



Figure 1: The framework of the proposed Meta Loss Correction (MLC) approach, which contains three stages: Virtual-Train (First), Meta-Train (Second) and Actual-Train (Third). GD means gradient descent algorithm.

$T^{t+1}$ which has a low validation loss. Since $D_v$ is clean, this supervision signal is ideal to guide the optimization of $T^{t+1}$. Note that the idea of 'validation' guided approach has also been used for model transfer learning via meta-learning [5] and differentiable neural network search [23].

During *Actual-Train*, the unrolled network weights $\theta^t$ are optimized to obtain $\theta^{t+1}$ with the updated $T^{t+1}$. Clearly, the *Actual-Train* is the step of 'Actual' backbone network optimization from unrolled network weights rather than a 'virtue' step. The optimization framework is shown in Fig. 1. Then we detail these three optimization steps separately.

**Virtual-Train** Given the noisy training set $D_\eta$, in each mini-batch, we fix $T^t$ and optimize the network weights $\theta^t$, thus the loss function at step $t$ is:

$$l_{virtual-trn} = -\frac{1}{n}\sum_{i=1}^{n}\tilde{y}_i \log(T^t f(x_i;\theta^t)) \quad (3)$$

where $n$ is the batch size in training set. Then the one-step-forward 'virtual' model weights $\hat{\theta}^{t+1}$ are optimized via gradient descent with learning rate $\alpha$:

$$\hat{\theta}^{t+1}(T^t) = \theta^t - \alpha\nabla_{\theta^t} l_{virtual-trn} \quad (4)$$

**Meta-Train** Given the one-step-forward backbone network (fixed $\hat{\theta}^{t+1}$), we can optimize the optimal $T^{t+1}$ on the validation set:

$$l_{meta-trn} = -\frac{1}{M}\sum_{i=1}^{M} y_i \log(f(x_i;\hat{\theta}^{t+1})) \quad (5)$$

However, this is still time and memory consuming, so we get an approximate estimate on a mini-batch of validation set:

$$l_{meta-trn} = -\frac{1}{m}\sum_{i=1}^{m} y_i \log(f(x_i; \hat{\theta}^{t+1})) \tag{6}$$

where m is the size of mini-batch. The transition matrix $T$ is also updated via gradient descent method with learning rate $\beta$:

$$u^{t+1} = T^t - \beta \nabla_{T^t} l_{meta-trn} \tag{7}$$

Apply chain rule to Eq. (7):

$$u^{t+1} = T^t - \beta\{\nabla_{\hat{\theta}^{t+1}} l_{meta-trn}(-\alpha \nabla^2_{\theta^t, T^t} l_{virtual-trn})\} \tag{8}$$

Note that $u^{t+1}$ is the *raw* one-step-forward noise transition matrix. $u^{t+1}$ cannot work as the final noise transition matrix because the entries of $u^{t+1}$ are not always non-negative and $u^{t+1}$ is not normalized. Thus, we first make $u^{t+1}$ become non-negative via:

$$\tilde{T}^{t+1} = \max(u^{t+1}, 0) \tag{9}$$

For the $j$th row of $\tilde{T}^{t+1}$, i.e. $\tilde{T}_j^{t+1} = [\tilde{T}_{j1}^{t+1}, ..., \tilde{T}_{jC}^{t+1}]$ which indicates all the probabilities transited to class $j$, we then perform normalization on $\tilde{T}_j^{t+1}$ to achieve the final $T_j^{t+1}$:

$$T_j^{t+1} = \frac{\tilde{T}_j^{t+1}}{\sum \tilde{T}_j^{t+1} + \delta(\sum \tilde{T}_j^{t+1})}, \delta(a) = \begin{cases} 1, & if \quad a = 0 \\ 0, & if \quad a \neq 0 \end{cases} \tag{10}$$

where $\delta(\cdot)$ is used to avoid division by 0.

**Actual-Train** After the 'virtual' network optimization and the Meta-Train, we now conduct the 'actual' network optimization on noisy training set by keeping $T^{t+1}$ fixed. Then we can achieve new network weights $\theta^{t+1}$ via gradient descent with learning rate $\gamma$ on $D_\eta$:

$$\theta^{t+1} = \theta^t - \gamma \nabla_{\theta^t}(-\frac{1}{n}\sum_{i=1}^{n} \tilde{y}_i \log(T^{t+1} f(x_i; \theta^t))) \tag{11}$$

The whole optimization framework (Virtual-Train, Meta-Train and Actual-Train) is summarized in Algorithm 1.

**Analysis** To understand the influence of noisy labels, we first go through the normal (no corrupted labels) deep model training process. During the initial training stage (Stage I), the network quickly fits easy samples. After that the network learns to fit hard samples (Stage II). This process (Stage I and II) is detailed in [4]. For deep model training with noisy samples, we can see noisy labels do not really affect training too much in Stage I because noisy samples are clearly not easy. During Stage II, the network cannot distinguish the hard samples with correct labels and noisy samples with wrong labels because these two types of samples both produce large loss. The supervision signals from wrong labels can make the network over-fit to the noisy samples during Stage II. The introduce of transition matrix $T$ actually aims to reduce this overfitting in Stage II. The approach [28] uses prior knowledge learned from noisy data to estimate $T$. However, this estimation cannot guarantee the accuracy of $T$ since this estimation essentially stems from noisy training data. Then the following approach GLC [12] realizes the limitation of [28], and conducts the estimation on a clean validation set which can provide accurate supervision signal. However, GLC does not optimize $T$ directly with a proper loss function associated with deep model training. Instead, GLC heuristically stacks the mean predictions of each class to construct $T$. In this work, our MLC also uses a clean validation set to avoid wrong supervision signal for estimating $T$. Moreover, we directly optimize $T$ associated with deep model training with the loss function which targets the best accuracy on the clean validation set. clearly, our approach is data driven rather than prior knowledge driven.

**Algorithm 1:** Meta Loss Correction (MLC)

**Input**: Randomly initialized $\{\theta^t, T^t\}$, noisy training set $D_\eta$, clean validation set $D_v$, the number of iterations $I$

**for** $t = 1, ..., I$ **do**

  Virtual-Train: Optimize the 'virtual' network weights $\hat{\theta}^{t+1}$ on $D_\eta$ via Eq. (3-4)

  Meta-Train: Optimize the transition matrix $T^{t+1}$ on $D_v$ via Eq. (5-10)

  Actual-Train: Optimize the 'actual' network weights $\theta^{t+1}$ on $D_\eta$ via Eq. (11)

**end**

**Output**: Model $\theta^{I+1}$

# 4. Experiments

## 4.1. Experimental Settings

We evaluate our approach on four computer vision (CV) datasets: MNIST [3], CIFAR-10 [19], CIFAR-100 [19] and Clothing1M [35], one natural language processing (NLP) dataset: Twitter [8]. Note that the noises in Clothing1M [35] are from the real world. And the noises on other

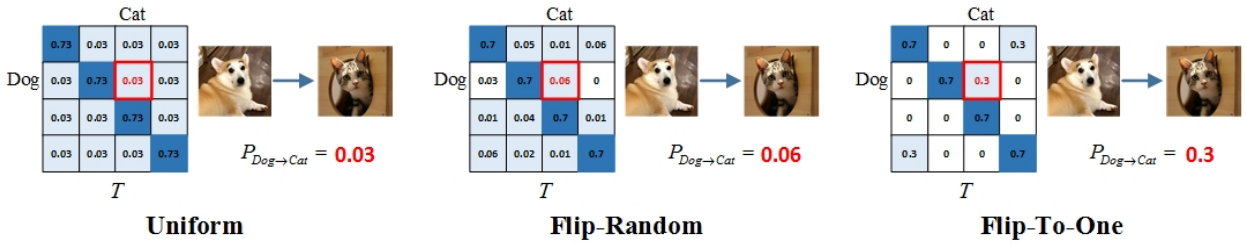



Figure 2: Visualization of three types of true noise transition matrix $T$ (30% noisy ratio). Using first 4 (10 classes in all) classes as an example.

datasets are artificially generated.

**MNIST** The MNIST [3] dataset is annotated with 10 object categories, containing $28 \times 28$ handwritten digit images. The training and test sets contain $60k$ and $10k$ images, respectively. For MNIST, we employ a network like LeNet [21] using SGD optimizer with learning rate $1e-2$. Set $\alpha = 1e-2, \beta = 1$ in this implementation.

**CIFAR** The CIFAR [19] dataset contains $32 \times 32$ color images. The training and test sets consist of $50K$ and $10K$ images, respectively. CIFAR-10 and CIFAR-100 contain 10 and 100 categories, respectively. Following [12], we use the Wide ResNet of depth 40 and widening factor 2 (WRN-40-2) [37] on these 2 datasets. We train the network using SGD optimizer with batch-size 64, learning rate $1e-4$, momentum 0.9 and weight decay $5e-4$. Set $\alpha = 1e-3, \beta = 1e-2$ in this implementation.

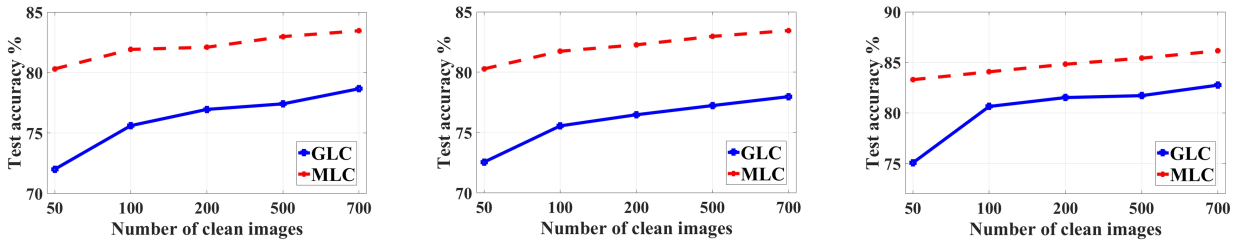
**Clothing1M** The Clothing1M dataset consists of $1M$ noisy data and additional $50K$, $14K$ and $10K$ clean data for training, validation and test sets, respectively. The Clothing1M dataset is annotated with 14 classes. Following [28], we use the ResNet-50 which is pre-trained on ImageNet with batch-size 32, learning rate $8e-3$, momentum 0.9 and weight decay $1e-3$. Set $\alpha = 1e-2, \beta = 1e-1$ in this implementation.

**Twitter** The Part-of-Speech Tagger for Twitter [8] dataset contains 1827 tweets annotated with 25 POS tags. The Twitter is split into one training set with 1000 tweets, one development set with 327 tweets, and one test set with 500 tweets. We merge the training and development sets to construct an augmented training set. Following [12], We use window size 3 and a two-layer fully connected network. We train the network using Adam optimizer with batch-size 64, learning rate $1e-3$ and weight decay $5e-5$. Set $\alpha = 1e-2, \beta = 1$ in this implementation.

**Noises** We conduct extensive experiments under different types of noise. Following [6, 30], we artificially corrupt the labels with three types of noise: flipping uniformly to all classes (Uniform), flipping randomly to any other class (Flip-Random) and flipping to one single different class (Flip-To-One). An example of noise transition matrix (30% noise ratio) under three types of noise is shown in Fig. 2. We evaluate our approach under different noise levels $\{10\%, 20\%, 30\%, 40\%\}$.

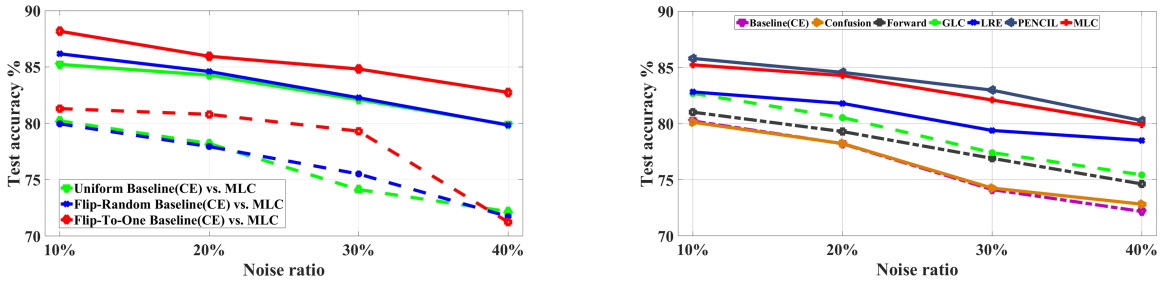
**Clean validation set** For CIFAR-10 and MNIST, we randomly sample 50 clean images per class, so $m = 500$. For CIFAR-100, we randomly sample 5 clean images per class, so $m = 500$. For Twitter, we sample 8 clean images per class, so $m = 200$.

**Compared approaches** We compare with state-of-the-art approaches [28, 12, 30, 10, 11] using the open-source implementations released by original papers. To make fair comparisons, all the approaches use the same training (noisy) and small validation (clean) sets. If the compared approaches do not rely on a small validation set, both the training and small validation sets are merged as training set. The compared approaches include: (1) Baseline (CE). We train a baseline model with CrossEntropy (CE) loss only (not using noise corrections at all). (2) Baseline (FC). We add an extra noise correction layer (Fully-Connected layer) at the end of the backbone network to simulate the noise transition matrix. (3) Forward Loss Correction (Forward). Forward [28] approximates $T$ using the maximum softmax probabilities of corresponding class from the noisy training dataset. (4) Gold Loss Correction (GLC). GLC [12] estimates $T$ using the mean prediction of all samples belonging to the same class from a small clean validation set. (5) Confusion Matrix. It is a simplified version of GLC that estimates $T$ by a confusion matrix [12]. (6) Learning to Reweight Examples (LRE). Instead of estimating $T$, LRE [30] learns to weight samples with the expectation that the noisy samples have small weights. (7) Co-teaching. Two deep neural networks are trained simultaneously to select training samples with small loss (expected clean data) for each other [11]. In this way, these two networks can teach each other by feeding (expected) clean samples for training. (8) Masking. Masking [10] proposes a structure-aware probabilistic model, which incorporates a (human-assisted) structure prior, to learn the noise transition probabilities. (9)PENCIL. PENCIL [36] introduces a probabilistic model, which can update both network parameters and label esti-




(a) Uniform (b) Flip-Random (c) Flip-To-One

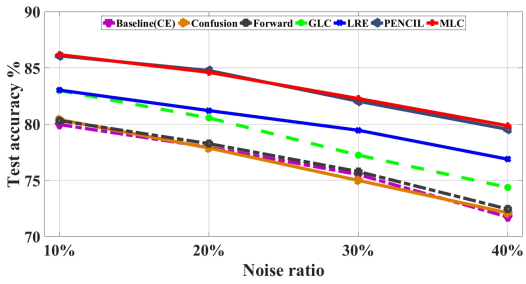
Figure 3: Test accuracy (%) on various sizes of clean image on CIFAR-10 under 30% noise ratio (a) Uniform noise; (b) Flip-Random noise; (c) Flip-To-One noise.



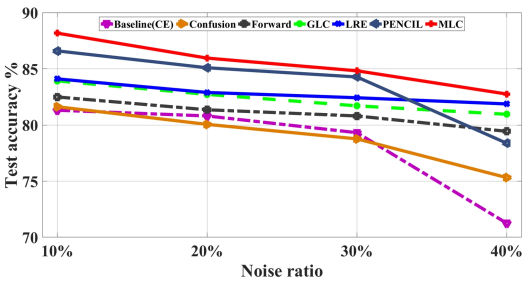

(a) CIFAR-10, All noise types




(b) CIFAR-10, Uniform




(c) CIFAR-10, Flip-Random




(d) CIFAR-10, Flip-To-One

Figure 4: The comparisons under various noise types and ratios. (a) Our MLC vs. Baseline(CE) under various types of noise; (b) Our MLC vs. state-of-the-art under Uniform noise; (c) Flip-Random noise; (d) Flip-To-One noise.

mations as label distributions.

### 4.2. Results

**The effect of the small clean validation set** Following [12, 30, 10, 11], we introduce a small clean validation set to estimate $T$. In our MLC framework, $T$ is optimized under the supervision of the loss on the small validation set in the stage of Meta-Train. Here we explore the influence of small validation set on the final noise-robust classification performance. We fix the training and test sets and change the size of small validation set. From Fig. 3, we can see that the increase of small validation set (clean images) can boost the performance. However, when the size of small validation set is larger than 100, the gain of performance is small. It means we do not need to annotate a large number of clean dataset to guide the meta training, which is very favorable in the real world. In addition, our MLC consistently outperforms our competitor GLC [12]. In particular, for an extremely small validation set (50 images), MLC significantly works better than GLC. It shows that our MLC is very robust to noises even when very small annotated clean samples are available.

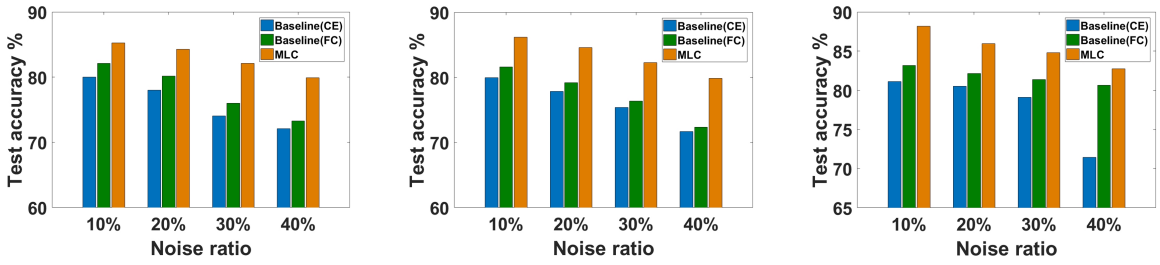
**Robustness to various types of noise** We explore the robustness of our MLC under various types of noise. Specifically, we test under 3 types of noise (Uniform, Flip-Random, Flip-To-One) with noise ratios {10%, 20%, 30%, 40%}. We make this evaluation on CIFAR-10 dataset. Fig. 4 (a) compares our MLC with Baseline(CE) approach.

Table 1: Test accuracy (%) on various datasets under various noise ratios. The best accuracy is in bold. Note that the mean accuracy across three types of noise and four ratios. CIFAR-100 cannot generate samples with 'Uniform' noise due to the limited ($< 100$) samples per class. The results of Co-teaching are copied from [11].

| Dataset | Method | Uniform | | | | Flip-Random | | | | Flip-To-One | | | | Mean |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 10% | 20% | 30% | 40% | 10% | 20% | 30% | 40% | 10% | 20% | 30% | 40% | |
| MNIST | Baseline(CE) | 98.59 | 98.39 | 98.27 | 98.02 | 98.60 | 98.46 | 98.25 | 98.18 | 97.59 | 96.65 | 94.26 | 84.68 | 96.66 |
| | Confusion[12] | 98.26 | 96.05 | 91.62 | 74.43 | 98.73 | 98.53 | 98.46 | 98.39 | 98.26 | 96.05 | 91.62 | 74.43 | 92.90 |
| | Froward[28] | 98.64 | 98.40 | 98.36 | 98.14 | 98.70 | 98.48 | 98.44 | 98.31 | 97.84 | 97.07 | 95.40 | 95.21 | 97.75 |
| | GLC[12] | 98.72 | 98.52 | 98.45 | 98.23 | 98.78 | 98.70 | 98.42 | 98.29 | 97.86 | 97.45 | 96.53 | 95.47 | 97.95 |
| | LRE[30] | 98.66 | 98.60 | 98.28 | 97.79 | 98.92 | 98.54 | 98.33 | 97.77 | 98.82 | 98.48 | 98.15 | **97.78** | 98.34 |
| | Co-teaching[11] | - | 97.25 | - | - | - | - | - | - | - | - | - | 87.63 | 97.25 |
| | MLC | **98.98** | **98.80** | **98.64** | **98.47** | **98.97** | **98.91** | **98.63** | **98.54** | **99.18** | **98.94** | **98.31** | 97.36 | **98.64** |
| CIFAR-10 | Baseline(CE) | 80.23 | 78.21 | 74.13 | 72.18 | 79.96 | 77.94 | 75.51 | 71.74 | 81.31 | 80.80 | 79.31 | 71.25 | 76.88 |
| | Confusion[12] | 80.12 | 78.23 | 74.26 | 72.83 | 80.40 | 77.87 | 74.99 | 72.11 | 81.62 | 80.05 | 78.76 | 75.31 | 77.21 |
| | Froward[28] | 81.02 | 79.29 | 76.91 | 74.63 | 80.31 | 78.26 | 75.78 | 72.44 | 82.49 | 81.35 | 80.80 | 79.43 | 78.56 |
| | GLC[12] | 82.69 | 80.54 | 77.42 | 75.44 | 82.98 | 80.55 | 77.24 | 74.37 | 83.92 | 82.72 | 81.70 | 80.95 | 80.04 |
| | LRE[30] | 82.82 | 81.80 | 79.39 | 78.51 | 83.02 | 81.20 | 79.45 | 76.88 | 84.10 | 82.89 | 82.42 | 81.87 | 81.20 |
| | Co-teaching[11] | - | 82.32 | - | - | - | - | - | - | - | - | - | 72.62 | 76.32 |
| | PENCIL[36] | **85.80** | **84.56** | **82.98** | **80.27** | 86.07 | **84.76** | 82.03 | 79.53 | 86.59 | 85.09 | 84.27 | 78.37 | 83.36 |
| | MLC | 85.23 | 84.28 | 82.10 | 79.89 | **86.17** | 84.60 | **82.27** | **79.85** | **88.17** | **85.95** | **84.82** | **82.75** | **83.84** |
| CIFAR-100 | Baseline(CE) | - | - | - | - | 50.67 | 45.18 | 41.68 | 37.40 | 52.98 | 47.72 | 44.19 | 37.78 | 44.56 |
| | Confusion[12] | - | - | - | - | 36.82 | 28.12 | 24.07 | 19.17 | 39.24 | 36.02 | 35.53 | 29.47 | 31.06 |
| | Forward[28] | - | - | - | - | 50.14 | 42.19 | 37.72 | 31.70 | 54.51 | 53.26 | 50.84 | 45.42 | 45.72 |
| | GLC[12] | - | - | - | - | 39.46 | 37.30 | 31.34 | 27.51 | 45.20 | 43.53 | 40.18 | 37.28 | 37.73 |
| | LRE[30] | - | - | - | - | 54.46 | 52.07 | 48.64 | 44.10 | 58.10 | 55.53 | 53.62 | 50.42 | 52.12 |
| | Co-teaching[11] | - | - | - | - | - | 54.23 | - | - | - | - | - | 34.81 | 43.47 |
| | PENCIL[36] | - | - | - | - | 59.97 | 56.15 | 51.75 | 44.85 | **60.03** | **58.48** | **57.33** | **52.62** | 55.15 |
| | MLC | - | - | - | - | **60.88** | **57.22** | **55.68** | **53.33** | 58.73 | 55.70 | 52.56 | 50.11 | **55.53** |
| Twitter | Baseline(CE) | **87.36** | 86.52 | **86.25** | 85.41 | 87.54 | **86.77** | 86.03 | **85.80** | 86.85 | 85.01 | 81.02 | 70.28 | 84.57 |
| | Confusion[12] | 84.75 | 86.36 | 85.75 | 85.47 | 86.84 | 85.65 | 85.85 | 84.49 | 86.06 | 85.97 | 84.08 | 79.65 | 85.08 |
| | Froward[28] | 79.05 | 77.86 | 81.80 | 78.38 | 55.01 | 75.39 | 79.32 | 68.88 | 47.17 | 65.30 | 71.38 | 58.32 | 69.82 |
| | GLC[12] | 87.13 | 86.40 | 85.94 | 85.31 | 86.88 | 85.69 | 85.29 | 84.31 | 86.40 | 86.05 | 85.65 | **85.54** | 85.88 |
| | LRE[30] | 86.73 | 86.26 | 85.75 | 85.20 | 86.29 | 85.44 | 85.30 | 84.10 | 86.60 | 86.07 | 85.62 | 84.91 | 85.69 |
| | MLC | 87.28 | **86.92** | 86.10 | **85.52** | **87.60** | 86.73 | **86.12** | 85.43 | **87.45** | **87.01** | **85.71** | 84.36 | **86.35** |

We can see that our MCL consistently outperforms Baseline(CE) across all the noise ratios and types, showing the effectiveness of our loss correction strategy. Fig. 4 (b) (c) (d) compare MLC with state-of-the-art approaches under 3 types of noise: Uniform, Flip-Random and Flip-To-One, respectively. Clearly, our MLC consistently outperforms the other approaches. In particular, our MLC works better than other Loss Correction approaches by a very large margin.

**The effect of meta-learning** It is interesting to explore the promising performance coming from our meta-learning or the clean validation set. Then we introduce Baseline (FC) which adds a Fully-Connected layer at the end of the backbone network to simulate the noise transition matrix. We use the clean validation set and an alternating optimization: optimizing the backbone network using the validation set and optimizing the backbone and FC layer using the noisy training set. From Fig. 5, Baseline (FC) works better than Baseline (CE), showing the usefulness of the noise




(a) Uniform (b) Flip-Random (c) Flip-To-One

Figure 5: Comparisons among MLC, Baseline(FC) and Baseline(CE) on CIFAR-10 under various noise types: (a) Uniform noise; (b) Flip-Random noise; (c) Flip-To-One noise.

transition matrix. Since both Baseline (FC) and MLC use the clean validation set, MLC outperforms Baseline (FC), showing the effectiveness of meta-learning.

**Comparisons with state-of-the-art** We make extensive comparisons (various noise types and ratios) with many state-of-the-art approaches on three popular computer vision (CV) datasets (MNIST, CIFAR-10 and CIFAR-100) and one natural language processing (NLP) dataset (Twitter) in Table 1. (1) For MNIST, our MLC consistently outperforms the other approaches across 3 noises. (2) For CIFAR-10, our MLC consistently works better than other approaches, e.g. 83.84% of MLC vs. 83.36% of PENCIL (the 2nd best) in terms of mean accuracy across all the noise ratios and types. PENCIL achieves similar performance with our MLC under individual noise ratios. (3) For CIFAR-100 dataset, our MLC outperforms other approaches under Flip-Random noise and achieves similar performance with PENCIL under Flip-To-One noise. (4) Apart from CV datasets, we also evaluate our MLC on one NLP dataset Twitter to verify the generalization capacity of MLC. Unlike CV datasets, the Baseline(CE) approach achieves comparable performance with the loss correction approaches, e.g. it even outperforms the Forward in terms of mean accuracy: 84.57% (Baseline(CE)) vs. 69.82% (Forward). Our MLC achieves the best mean accuracy: 86.35% (MLC) vs. 85.88% (GLC, the 2nd best).

To summarize, MLC is very robust to noises (across various types and ratios) from CV to NLP tasks. In comparison, another state-of-the-art loss correction approach GLC works well on MNIST and CIFAR-10, however, the performance drops significantly on a more challenging dataset CIFAR-100.

**Real-world noises** Finally, to show the robustness of our approach under real-world noises, we test our MLC on Clothing1M dataset. As shown in Table 2, the results of *CrossEntropy* and *Forward* are copied from [28], *GLC* and *Mask* are copied from [12] and [10], respectively. We can see that our MLC and Mask [10] achieve the best performance, significantly outperforming other approaches. It means that our MLC and Mask [10] are both very robust to real-world noises. Note that Mask [10] manually defines the prior knowledge that which classes are similar (e.g. cat and dog) and which are not (e.g. cat and car). Then this human-defined prior knowledge is used to optimize the noise transition matrix $T$. In comparison, our MLC learns $T$ directly and automatically from data.

Table 2: Test accuracy (%) on Clothing1M.

| Approach | Prior knowledge | Accuracy |
|---|---|---|
| CrossEntropy [28] | no | 68.94 |
| Forward [28] | yes | 69.84 |
| GLC [12] | yes | 70.84 |
| Mask [10] | yes | **71.10** |
| MLC | no | 71.06 |

## 5. Conclusion

In this work, when a small clean dataset is available, we propose a learning-based loss correction approach, Meta Loss Correction (MLC), which can learn noise transition matrix $T$ and network weights jointly via meta-learning. Unlike most existing approaches which estimate $T$ using prior knowledge, MLC learns $T$ directly from data without 'perfect example' assumption and human-in-the-loop process. Extensive experiments are conducted on both CV and NLP datasets. Results show that our MLC approach compares favorably with other loss correction approaches and general state-of-the-art noise-robust deep learning approaches.

## Acknowledgment

This work is supported by National Key Research and Development Project under Grant 2019YFB2101901 and National Natural Science Foundation of China under Grants 61925602 and 61732011.

## References


[1] Alan Joseph Bekker and Jacob Goldberger. Training deep neural-networks based on unreliable labels. In *ICASSP*, pages 2682–2686. IEEE, 2016.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[3] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[4] Yang Fan, Fei Tian, Tao Qin, Jiang Bian, and Tie-Yan Liu. Learning what data to learn. *arXiv preprint arXiv:1702.08635*, 2017.

[5] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pages 1126–1135. JMLR. org, 2017.

[6] Aritra Ghosh, Himanshu Kumar, and PS Sastry. Robust loss functions under label noise for deep neural networks. In *AAAI*, pages 1919–1925, 2017.

[7] Aritra Ghosh, Naresh Manwani, and PS Sastry. Making risk minimization tolerant to label noise. *Neurocomputing*, 160:93–107, 2015.

[8] Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. Part-of-speech tagging for twitter: Annotation, features, and experiments. Technical report, Carnegie-Mellon Univ Pittsburgh Pa School of Computer Science, 2010.

[9] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[10] Bo Han, Jiangchao Yao, Gang Niu, Mingyuan Zhou, Ivor Tsang, Ya Zhang, and Masashi Sugiyama. Masking: A new perspective of noisy supervision. In *Advances in Neural Information Processing Systems*, 2018.

[11] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems*, 2018.

[12] Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. Using trusted data to train deep networks on labels corrupted by severe noise. *NIPS*, 2018.

[13] Guosheng Hu, Yang Hua, Yang Yuan, Zhihong Zhang, Zheng Lu, Sankha S Mukherjee, Timothy M Hospedales, Neil M Robertson, and Yongxin Yang. Attribute-enhanced face recognition with neural tensor fusion networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3744–3753, 2017.

[14] Guosheng Hu, Yongxin Yang, Dong Yi, Josef Kittler, William Christmas, Stan Z Li, and Timothy Hospedales. When face recognition meets with deep learning: an evaluation of convolutional neural networks for face recognition. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 142–150, 2015.

[15] Lu Jiang, Deyu Meng, Shoou-I Yu, Zhenzhong Lan, Shiguang Shan, and Alexander Hauptmann. Self-paced learning with diversity. In *Advances in Neural Information Processing Systems*, pages 2078–2086, 2014.

[16] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Regularizing very deep neural networks on corrupted labels. *ICML*, 2017.

[17] Ishan Jindal, Matthew Nokleby, and Xuewen Chen. Learning deep networks from noisy labels with dropout regularization. In *ICDM*, pages 967–972. IEEE, 2016.

[18] Donald R Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21(4):345–383, 2001.

[19] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[21] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[22] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *ICCV*, pages 1928–1936, 2017.

[23] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[24] Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE TPAMI*, 38(3):447–461, 2016.

[25] Xin Liu, Shaoxin Li, Meina Kan, Shiguang Shan, and Xilin Chen. Self-error-correcting convolutional neural network for learning with noisy labels. In *ICFG 2017*, pages 111–117. IEEE, 2017.

[26] Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *ICML-2015*, pages 2113–2122, 2015.

[27] Volodymyr Mnih and Geoffrey E Hinton. Learning to label aerial images from noisy data. In *ICML-12*, pages 567–574, 2012.

[28] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, pages 2233–2241, 2017.

[29] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

[30] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. *arXiv preprint arXiv:1803.09050*, 2018.

[31] Sainbayar Sukhbaatar and Rob Fergus. Learning from noisy labels with deep neural networks. *arXiv preprint arXiv:1406.2080*, 2(3):4, 2014.

[32] Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge J Belongie. Learning from noisy large-scale datasets with minimal supervision. In *CVPR*, pages 6575–6583, 2017.

[33] Yisen Wang, Weiyang Liu, Xingjun Ma, James Bailey, Hongyuan Zha, Le Song, and Shu-Tao Xia. Iterative learning with open-set noisy labels. *arXiv preprint arXiv:1804.00092*, 2018.

[34] Xiang Wu, Ran He, Zhenan Sun, and Tieniu Tan. A light cnn for deep face representation with noisy labels. *IEEE Transactions on Information Forensics and Security*, 13(11), 2018.

[35] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *CVPR*, pages 2691–2699, 2015.

[36] Kun Yi and Jianxin Wu. Probabilistic end-to-end noise correction for learning with noisy labels. *CVPR2019*, 2019.

[37] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[38] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[39] Dingwen Zhang, Deyu Meng, and Junwei Han. Co-saliency detection via a self-paced multiple-instance learning framework. *IEEE transactions on pattern analysis and machine intelligence*, 39(5):865–878, 2017.

[40] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *NIPS*, pages 8778–8788, 2018.

[41] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.